



keyestudio

www.keyestudio.com

Getting Started with Mixly





Guide Content

1. Introduction for Mixly.....	4
2. Design Concept and User Groups.....	6
2.1 Design Concept.....	6
2.2 User Groups.....	8
3. Interface Functions of Mixly.....	9
3.1 System Functions.....	9
3.2 In/Out Block.....	11
3.3 Control Block.....	16
3.4 Math Block.....	22
3.5 Text Block.....	27



keyestudio

www.keyestudio.com

3.6 List Block.....	30
3.7 Logic Block.....	33
3.8 Variable Block.....	36
3.9 SerialPort Block.....	39
3.10 Communicate Block.....	45
3.11 Sensor Block.....	49
3.12 Actuator Block.....	52
3.13 Monitor Block.....	55
3.14 Functions Block.....	59
4. Resources.....	64

1. Introduction for Mixly

Mixly is a free open-source graphical Arduino programming software, based on Google's Blockly graphical programming framework, and developed by Mixly Team@ BNU.

It is a free open-source graphical programming tool for creative electronic development; a complete support ecosystem for creative e-education; a stage for maker educators to realize their dreams.

Although there is an Ardublock graphical programming software launched by Arduino official, Ardublock is not perfect enough, and many common functions cannot be realized.

The figure below shows the functional comparison between Ardublock and Mixly.

	If/ else	for / while loop	Math & Boolean	Bluetooth	IR	Interrupt	Serial Communicate
Ardublock	✓	✓	✓	✓	×	×	×
Mixly	✓	✓	✓	✓	✓	✓	✓
	Lists	Pulse	ShiftOut	Variables Declare		Procedure	
Ardublock	×	×	×	only support Integer/ Float		Not return data	
Mixly	✓	✓	✓	support Integer/ Float /Char/ Boole		Return data	

It can be said that Mixly is the most versatile and smoothest Arduino graphical programming software, which can replace the Arduino programming tool IDE.

2. Design Concept and User Groups

2.1 Design Concept

(1) Usability

Mixly is designed to be completely green. Currently Mixly supports win, ubuntu, mac. Windows users can download the Mixly package directly from the Internet, and unzip it to run on Windows XP and above (download link is attached below).

(2) Simplicity

Mixly uses the Blockly graphical programming engine to replace complex text manipulation with graphical building blocks, providing a good foundation for beginners to get started quickly.

- ① Use the different color icons to represent different types of functional blocks, very convenient for users to classify.
- ② Provide default options in the composite function block to effectively reduce the number of user drags.
- ③ Integrate all the features of the software in the same interface.

- ④ Provide the reference tutorial and code examples.

(3) Functionality

It has versatile functions. Mixly can almost implement all the functions that Arduino IDE has. Support all official development boards of arduino.

(4) Continuity

The goal of the graphical programming system is definitely not to replace the original text programming method, but to better understand the programming principles and program thinking through graphical programming, and lay the foundation for future text programming.

It is also the design philosophy for Mixly. More continuous content has been added to the design of the software to protect the user's learning outcomes. To be specific, it includes the introduction of variable types, the consistency of text programming as much as possible in the design of the module, and the support of both graphical and text programming.

(5) Ecological

The most important design concept of Mixly is its ecological feature, which can distinguish it from other

Arduino graphical programming.

In order to achieve sustainable development, Mixly is designed to allow manufacturers to develop their own unique modules (currently supports DfRobot, StartLab, MakeBlock, Sense, Seeed, Lubot. But users require JavaScript programming foundation to make this part of the module).

It also allows users directly use Mixly's graphical programming function to generate common modules (such as LED digital display, buzzer broadcast, etc. Users are able to make this part of the module only using Mixly).

Both of the two kinds of modules mentioned above can be imported into the Mixly system through the "Import" function, thereby realizing the user's own value in the popularity of Mixly software.

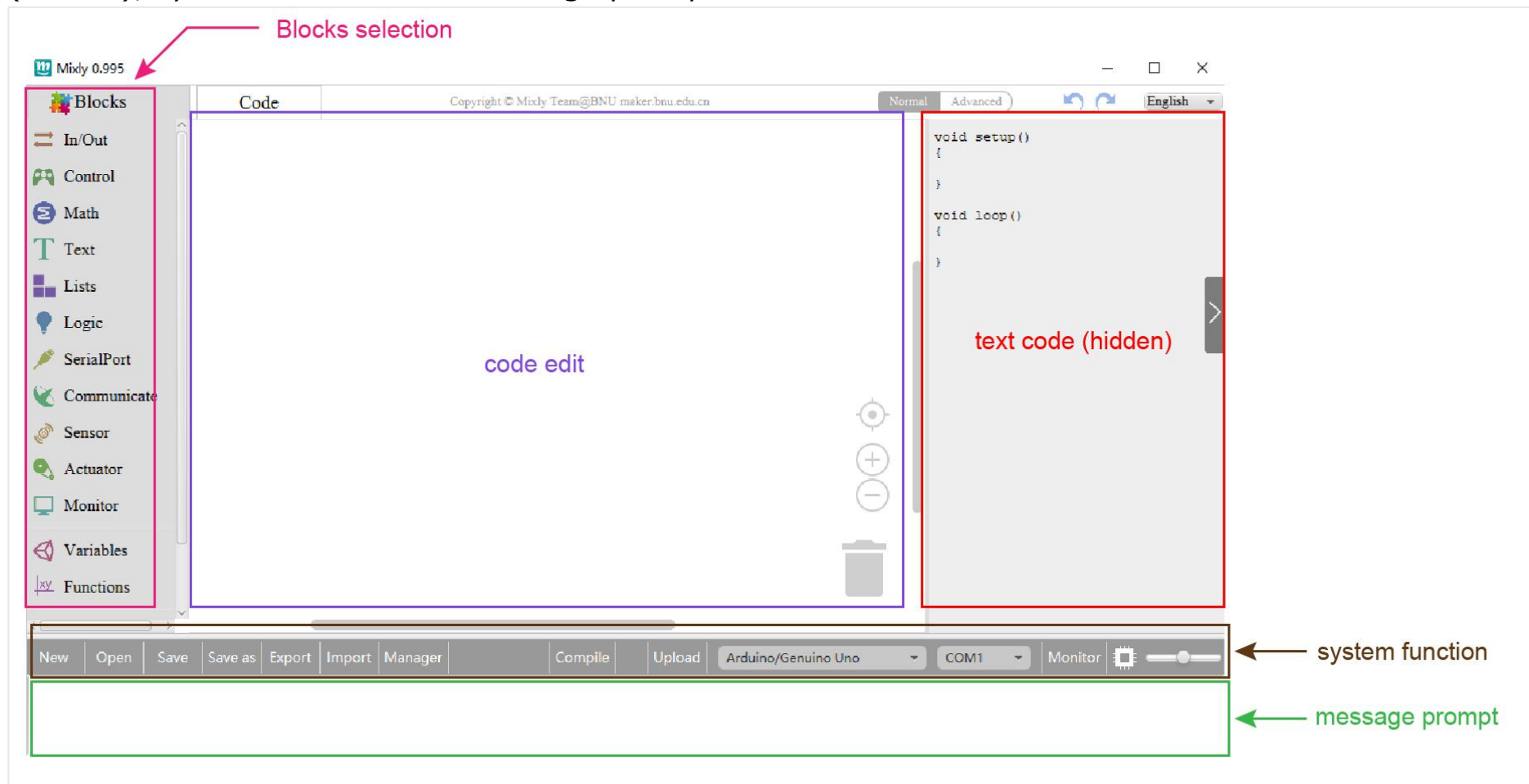
2.2 User Groups

From the above design concept, it can be seen that Mixly is suitable for primary and secondary school students to cultivate programming thinking. It is also available for quick programming when creating a work. Of course, it is good for those lovely friends who don't want to learn text programming, but want to do some small works with intelligent control.

3. Interface Functions of Mixly

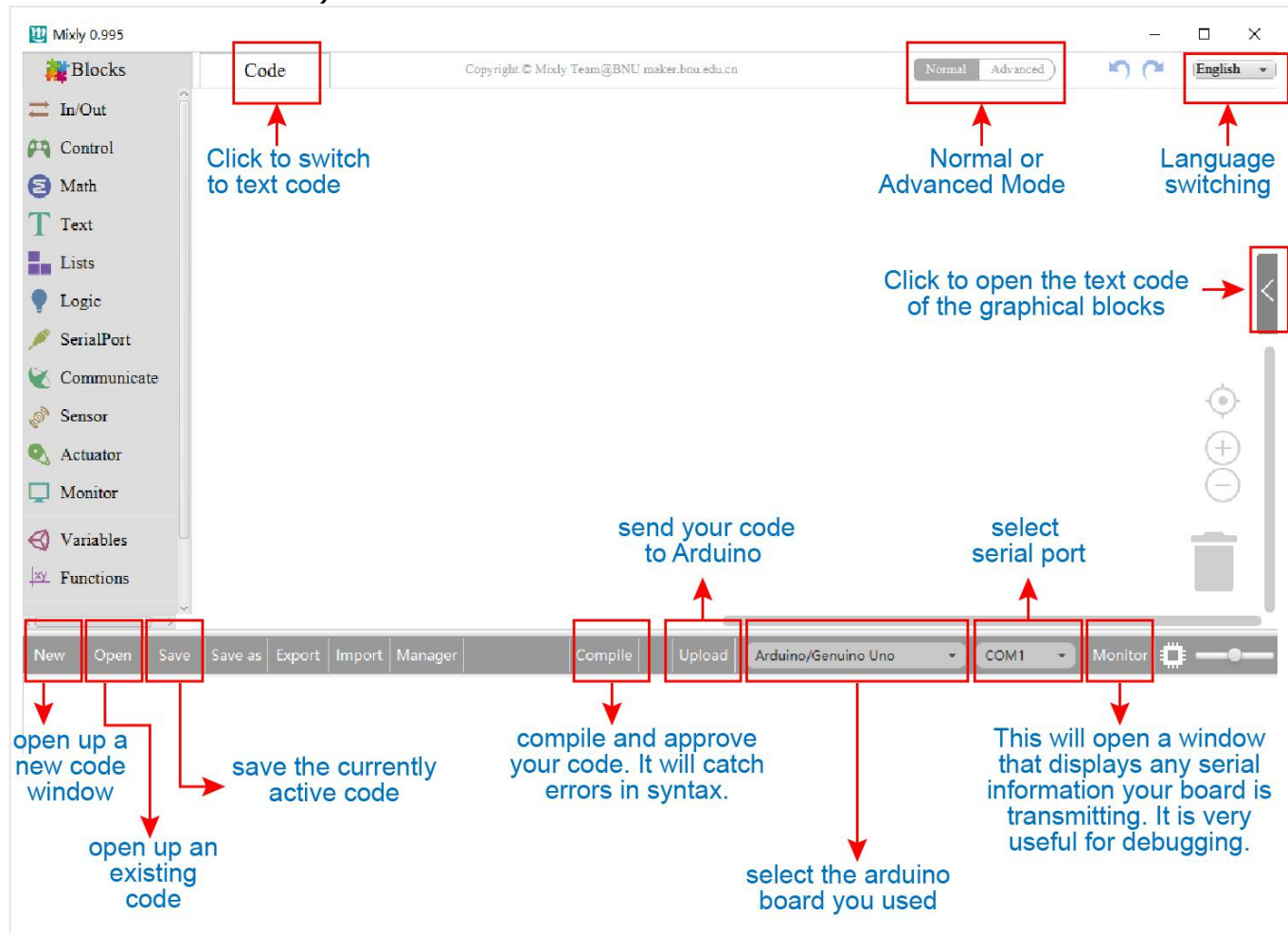
3.1 System Functions

Look at the main interface of Mixly, it includes five parts, that is, Blocks selection, code edit, text code (hidden), system function and message prompt area. Shown below.



Some common functions:

Through this interface, you can complete the code compile、upload、save and manage. It support four remove methods: drag it left out code window, or drag to Recycle Bin, delete key, or right-click to delete block. It supports four languages: English、Español (Spanish)、中文简体(Chinese Simplified)、中文繁体 (Chinese Traditional).



3.2 In/Out Block

The screenshot displays the Arduino IDE's Blocks editor. On the left, a sidebar lists block categories: In/Out, Control, Math, Text, Lists, Logic, SerialPort, Communicate, Sensor, Actuator, Monitor, Variables, Functions, CarControl, and saynum. The 'In/Out' category is selected, showing a list of blocks including DigitalWrite, DigitalRead, AnalogWrite, AnalogRead, attachInterrupt, detachInterrupt, pinMode, pulseIn, and ShiftOut. The main workspace contains a sequence of these blocks: a 'HIGH' block, a 'DigitalWrite PIN# 0 Stat HIGH' block, a 'DigitalRead PIN# 0' block, an 'AnalogWrite PIN# 3 value 0' block, an 'AnalogRead PIN# A0' block, an 'attachInterrupt pin# 2 mode RISING' block followed by a 'do' loop block, a 'detachInterrupt pin# 2' block, a 'pinMode 0 Stat INPUT' block, a 'pulseIn(μs) PIN# 0 state HIGH' block, a 'pulseIn(μs) PIN# 0 state HIGH timeout(μs) 1000000' block, and a 'ShiftOut dataPin# 0 clockPin# 0 bitOrder MSBFIRST data 0' block. The bottom status bar shows 'New', 'Open', 'Save', 'Save as', 'Export', 'Import', 'Manager', 'Compile', 'Upload', 'Arduino/Genuino Uno', and 'COM1'.

Blocks

Code

Copyright © Mixly Team@BNU maker.bnu.edu.cn

Normal Advanced

In/Out

Control

Math

Text

Lists

Logic

SerialPort

Communicate

Sensor

Actuator

Monitor

Variables

Functions

CarControl

saynum

HIGH

DigitalWrite PIN# 0 Stat HIGH

DigitalRead PIN# 0

AnalogWrite PIN# 3 value 0

AnalogRead PIN# A0

attachInterrupt pin# 2 mode RISING

do

detachInterrupt pin# 2

pinMode 0 Stat INPUT

pulseIn(μs) PIN# 0 state HIGH

pulseIn(μs) PIN# 0 state HIGH timeout(μs) 1000000

ShiftOut dataPin# 0 clockPin# 0 bitOrder MSBFIRST data 0

New Open Save Save as Export Import Manager Compile Upload Arduino/Genuino Uno COM1

NO.	BLOCK ICON	DEFINITION
1		Returns HIGH or LOW voltage
2		Write digital value to a specific Port. Digital Output: set the HIGH or LOW output for IO pins
3		Returns a digital value of a specific Port. Digital IO Read Pin, generally used to read the HIGH or LOW level detected by Digital sensor
4		Write analog value between 2 and 255 to a specific Port. Analog Output: set the Analog value output by Analog IO pins (0~255).

5		Returns value between 0 and 1023 of a specific Port. Analog IO Read Pin, generally used to read the Analog value detected by Analog sensor.
6		External Interrupts function, with three trigger interrupt modes RISING, FALLING, CHANGE.
7		Detachs interrupt to a specific Port. Turn off the given interrupt function.
8		Set the IO pins as Output or Input state

9		Read the continuous time of HIGH or LOW pulse from IO pins (generally used for ultrasonic ranging)
10		Read a pulse (either HIGH or LOW) on a pin within a time set in timeout.
11		Set the ShiftOut data pin, clock pin. Output the data needed from the bitOrder MSBFIRST or LSBFIRST (Most Significant Bit First, or, Least Significant Bit First). Generally used for controlling the 74HC595 CHIP.
12		This is the function interface under Normal mode. If select Advanced mode, the functions will be more.

For example:

Connect your Arduino Uno board, then follow the steps below to light the Pin13 led on Arduino UNO.

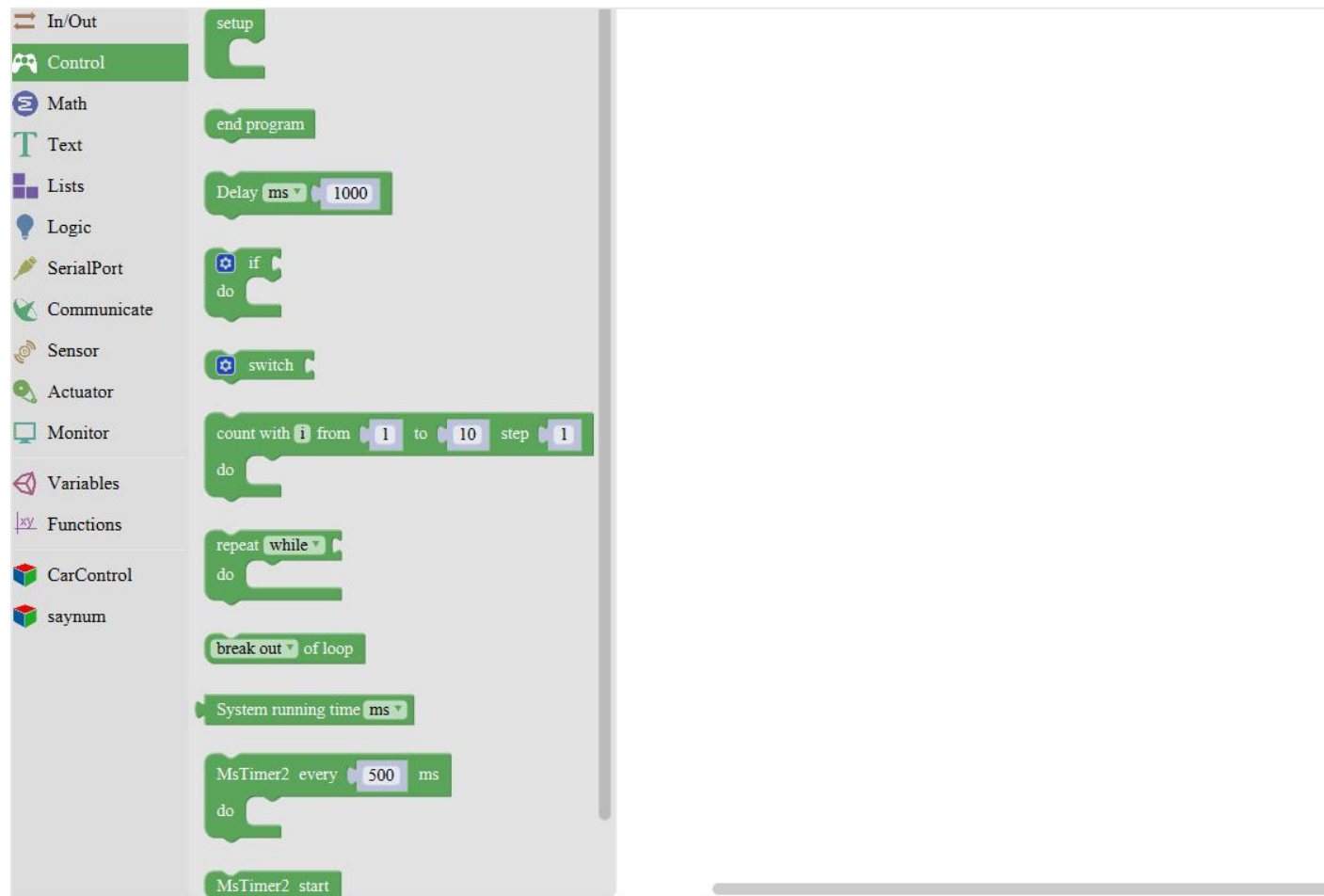
The screenshot shows the Arduino IDE interface with the following components and annotations:

- Blocks Panel (Left):** A list of block categories. The **In/Out** category is highlighted with a black box and labeled with an arrow and the number (1).
- Code Editor (Right):** Contains the following C++ code:

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13,HIGH);
}
```
- Block Editor (Center):** A **DigitalWrite** block is shown. The **PIN#** field is set to **13** (labeled with an arrow and the number 2), and the **Stat** field is set to **HIGH** (labeled with an arrow and the number 3).
- Toolbar (Bottom):** The **Upload** button is highlighted with a black box and labeled with an arrow and the number (6). The **Arduinio/Genuino Uno** dropdown menu is highlighted with a black box and labeled with an arrow and the number (4). The **COM8** dropdown menu is highlighted with a black box and labeled with an arrow and the number (5).
- Status Bar (Bottom):** Displays the message "Upload success!".

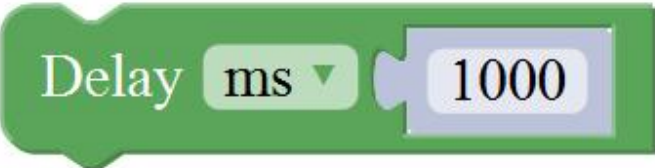
3.3 Control Block



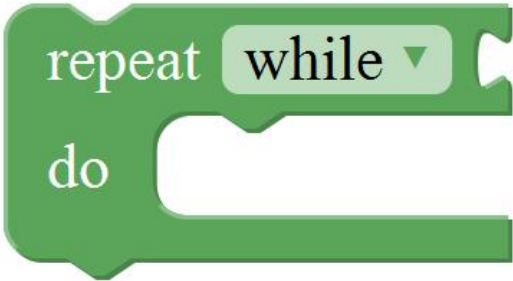
The image shows the Scratch Control block palette on the left and a workspace on the right. The palette contains the following blocks:

- Control** (selected):
 - setup
 - end program
 - Delay ms 1000
 - if do
 - switch
 - count with i from 1 to 10 step 1 do
 - repeat while do
 - break out of loop
 - System running time ms
 - MsTimer2 every 500 ms do
 - MsTimer2 start
- In/Out
- Math
- Text
- Lists
- Logic
- SerialPort
- Communicate
- Sensor
- Actuator
- Monitor
- Variables
- Functions
- CarControl
- saynum

The workspace on the right is empty.

NO.	BLOCK ICON	DEFINITION
1		Initialization (run only once)
2		End the program, means the program will stop running when use this block.
3		Delay function, click to select ms or us (pause the program for the amount of time (in milliseconds) specified as parameter. There are 1000 milliseconds in a second.)

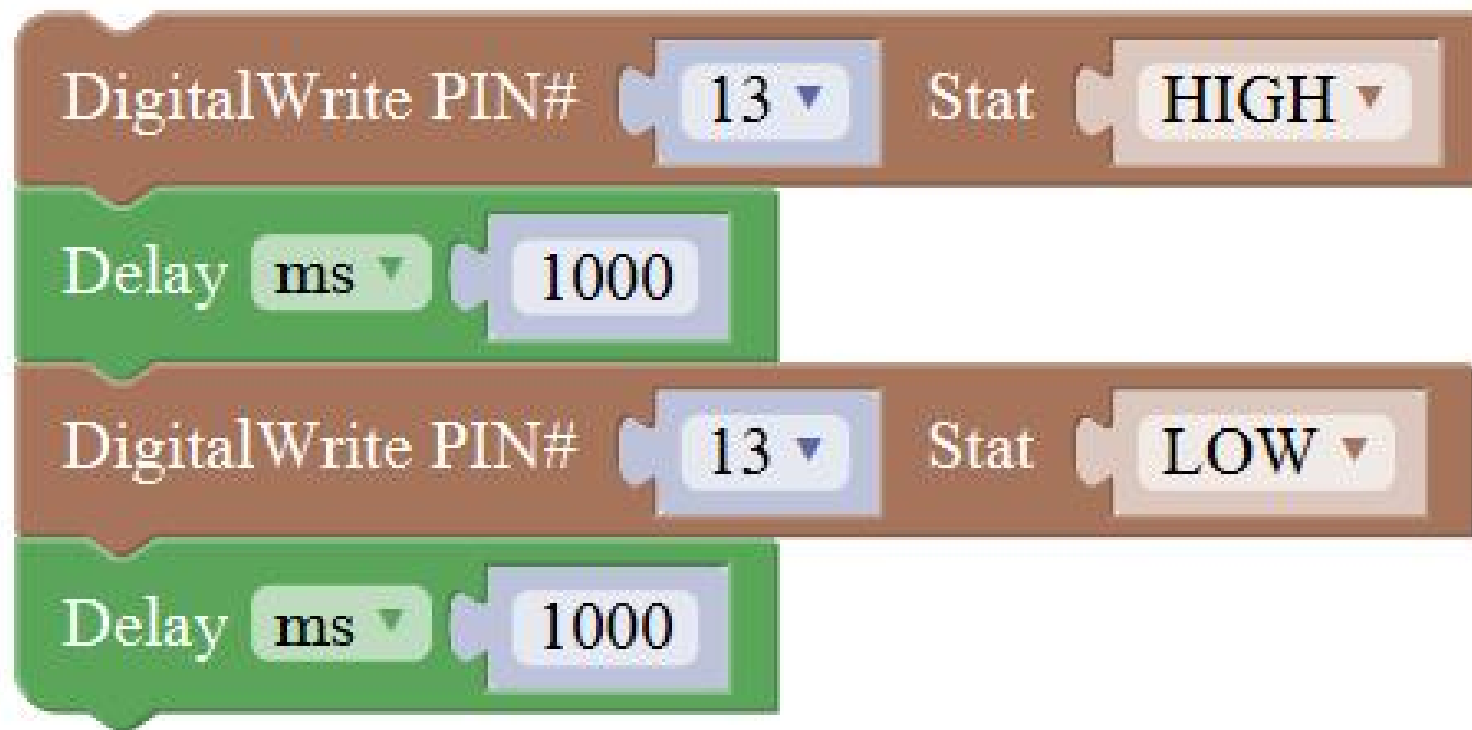
4		if_do function (first evaluate a value be <u>true or false</u>, if a value is true, then do some statement. You can click the blue gear icon to select the else if block or else block.)
5		switch function. You can click the blue gear icon to select the case block or default block. (used to evaluate several programs then execute the corresponding function matched with program.)
6		Equal to <u>for</u> statement.

7		A while loop statement.
8		break function, used to exit from the containing loop.
9		millis() function, returns the system running time since the program started. (The unit can be ms (milliseconds) or µs (microsecond)).

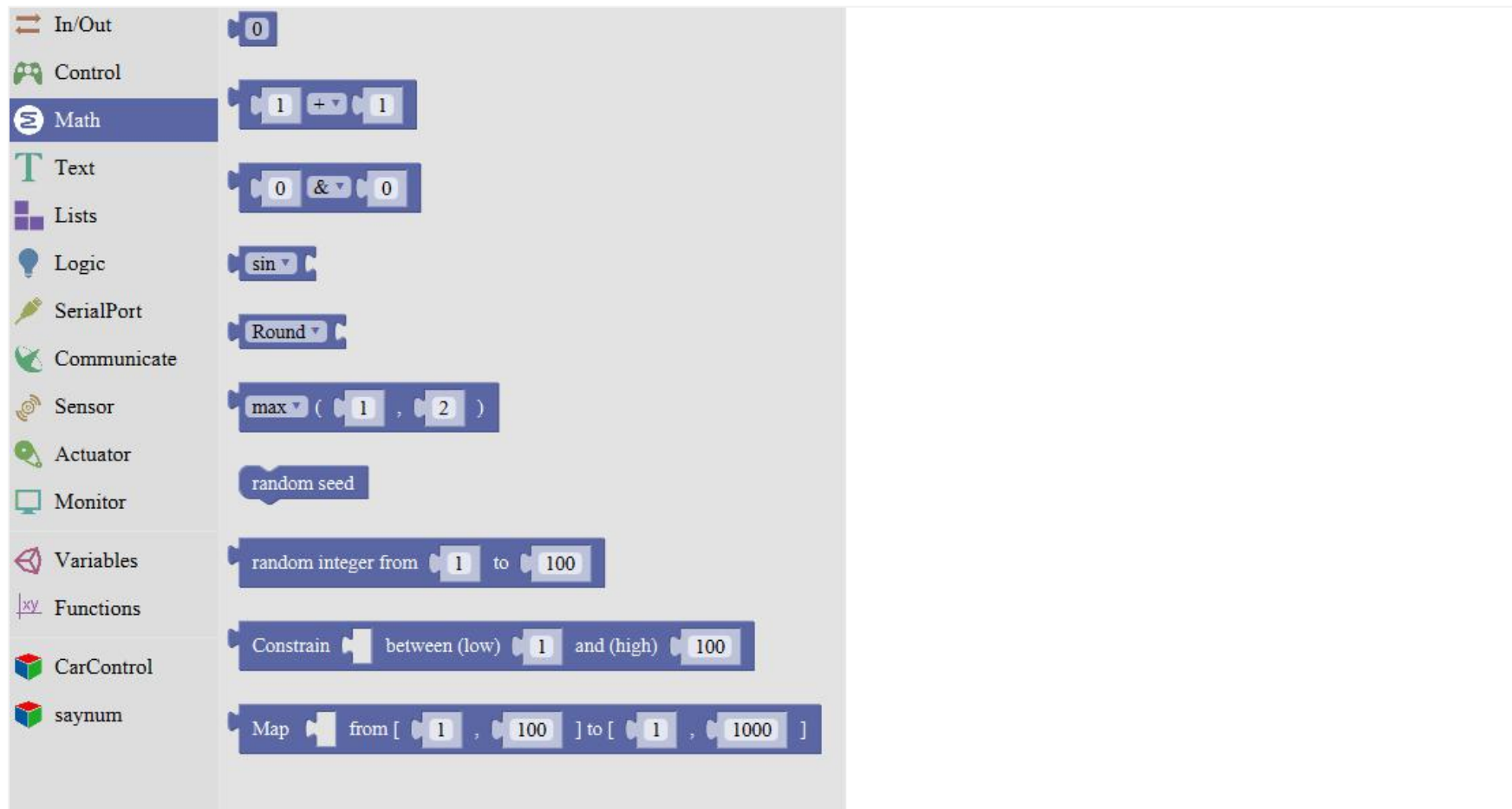
10	 A green Scratch block with a notch on the left and a bump on the right. It contains the text "MsTimer2 every 500 ms" and "do". The number "500" is inside a small grey box.	Timer interrupt function, that is, set a trigger interrupt for the amount of time (in milliseconds) specified as parameter.
11	 A green Scratch block with a notch on the left and a bump on the right. It contains the text "MsTimer2 start".	Timer interrupt start block
12	 A green Scratch block with a notch on the left and a bump on the right. It contains the text "MsTimer2 stop".	Timer interrupt stop block

For example:

Compile and upload the program below to your Arduino board, you should see Pin13 LED on Arduino UNO continue to flash.(with an interval of 1s, equal to 1000ms)



3.4 Math Block



The image displays the 'Math' block palette in a Scratch-like environment. The palette on the left lists various categories: In/Out, Control, Math (highlighted), Text, Lists, Logic, SerialPort, Communicate, Sensor, Actuator, Monitor, Variables, Functions, CarControl, and saynum. The main workspace shows several Math blocks: a constant '0', an addition block (1 + 1), a multiplication block (0 & 0), a 'sin' function block, a 'Round' block, a 'max' block with inputs 1 and 2, a 'random seed' block, a 'random integer from' block with inputs 1 and 100, a 'Constrain' block with inputs 1 and 100, and a 'Map' block with inputs [1, 100] and [1, 1000].

Math Block Palette:

- In/Out
- Control
- Math**
- Text
- Lists
- Logic
- SerialPort
- Communicate
- Sensor
- Actuator
- Monitor
- Variables
- Functions
- CarControl
- saynum

Math Blocks in Workspace:

- 0
- 1 + 1
- 0 & 0
- sin
- Round
- max (1 , 2)
- random seed
- random integer from 1 to 100
- Constrain between (low) 1 and (high) 100
- Map from [1 , 100] to [1 , 1000]

NO.	BLOCK ICON	DEFINITION
1		A number
2		Click to select the Arithmetic Operators: <u>+</u> (addition); <u>-</u> (subtraction); <u>x</u> (Multiplication); <u>÷</u> (division); <u>%</u> (remainder); <u>^</u> (bitwise xor)
3		Click to select the <u>&</u> (bitwise and); <u> </u> (bitwise or); <u><<</u> (bitshift left); <u>>></u> (bitshift right)

4		Click to select the <u>sin</u>; <u>cos</u>; <u>tan</u>; <u>asin</u>; <u>acos</u>; <u>atan</u>; <u>ln</u>; <u>log10</u>; <u>e^</u>; <u>10^</u>; <u>++ (increment)</u> ; <u>-- (decrement)</u>
5		Click to select the Round; Ceil; Floor; <u>abs</u>; <u>sq</u>; <u>sqrt</u> Round: Returns the integer part a number using around. Ceil: Returns the integer part a number using ceil. Floor: Returns the integer part a number using floor. abs: Return the absolute value of a number. sq: Return the square of a number. sqrt: Return the square root of a number.

6		If select the max , returns the larger number; if select the min , returns the smaller number.
7		Initialize the random seed
8		Return a random integer between the two specified limits, inclusive.
9		Constrain a number to be between the specified limits (inclusive). (generally used to constrain an analog value read from sensor)

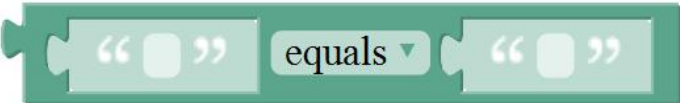
10	 A Scratch 'Map' block. The block is blue with a white 'Map' label and a small white icon. It contains the text 'from [1 , 100] to [1 , 1000]'. The numbers 1, 100, 1, and 1000 are each inside a light blue rounded rectangle.	Map a number from the first interval to the second interval. (For instance, potentiometer-controlled servo, map the range of potentiometer (0, 1023) to the angle of servo (0, 180)).
----	---	---

3.5 Text Block

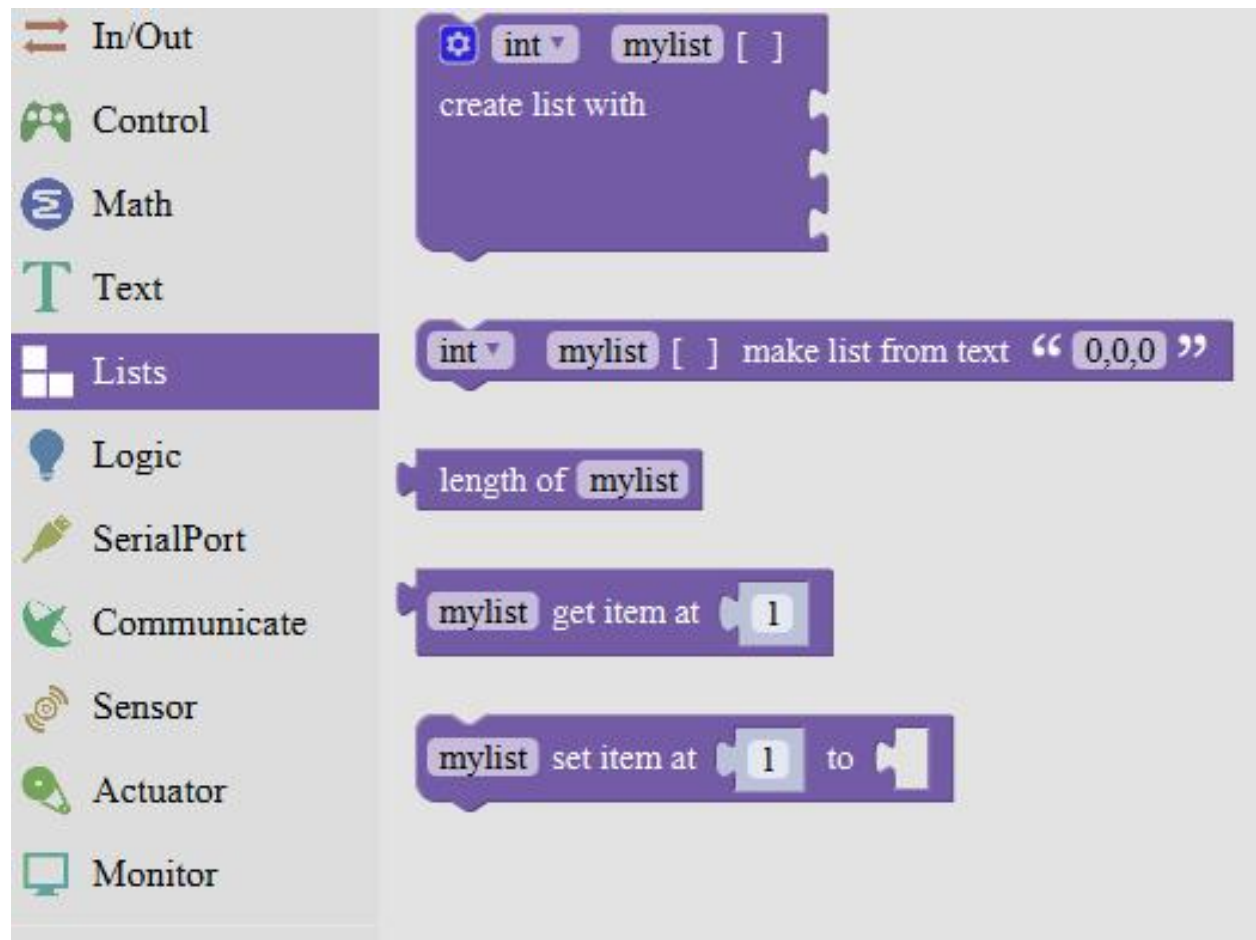
The image displays the 'Text' block palette in a Scratch-like environment. The palette is organized into categories on the left, with the 'Text' category highlighted in green. The categories and their corresponding blocks are as follows:

- In/Out**: A block with the text "hello".
- Control**: A block with the text "a".
- Math**: A block with the text "Hello" joined to "Mixly".
- Text** (highlighted):
 - A block with the text "Hello" joined to "Mixly".
 - A block with the text "123" converted to an integer (toInt).
 - A block with the number 223 converted to a character (toChar).
 - A block with the character 'a' converted to its ASCII value (toAscii).
 - A block with the number 0 converted to a string (toString).
 - A block with the text "hello" and a length of 5 (length of).
 - A block with the text "hello" and a character at index 0 (char at).
 - A block with two empty string boxes and an equals sign (equals).
 - A block with two empty string boxes and a compareTo function (compareTo).
- Lists**: No blocks are visible.
- Logic**: No blocks are visible.
- SerialPort**: No blocks are visible.
- Communicate**: No blocks are visible.
- Sensor**: No blocks are visible.
- Actuator**: No blocks are visible.
- Monitor**: No blocks are visible.
- Variables**: No blocks are visible.
- Functions**: No blocks are visible.
- CarControl**: No blocks are visible.
- saynum**: No blocks are visible.

NO.	BLOCK ICON	DEFINITION
1		character string: a letter, word, or line of text.
2		A character
3		Creates a piece of text by joining together two piece of text. (Here Hello join Mixly equals HelloMixly)
4		Converts a string into an integer or an float.
5		Returns the char corresponding to an ASCII code (Decimal number 97 corresponding to a)

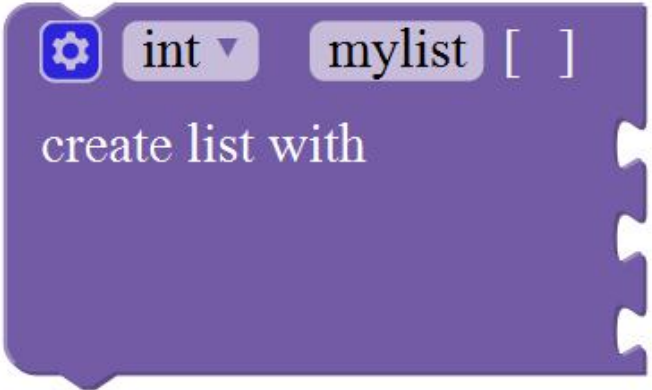
6		Returns the ASCII code corresponding to a char.
7		Converts a number into a string.
8		Calculates the length of a string
9		Output the char of a string (the char at 0 of hello is h)
10		The first string equals or startsWith or endsWith the second string, returns 1, otherwise returns 0. (if equals, both strings are abc, returns 1.)
11		Returns a decimal value of the first string subtracts the second string.

3.6 List Block



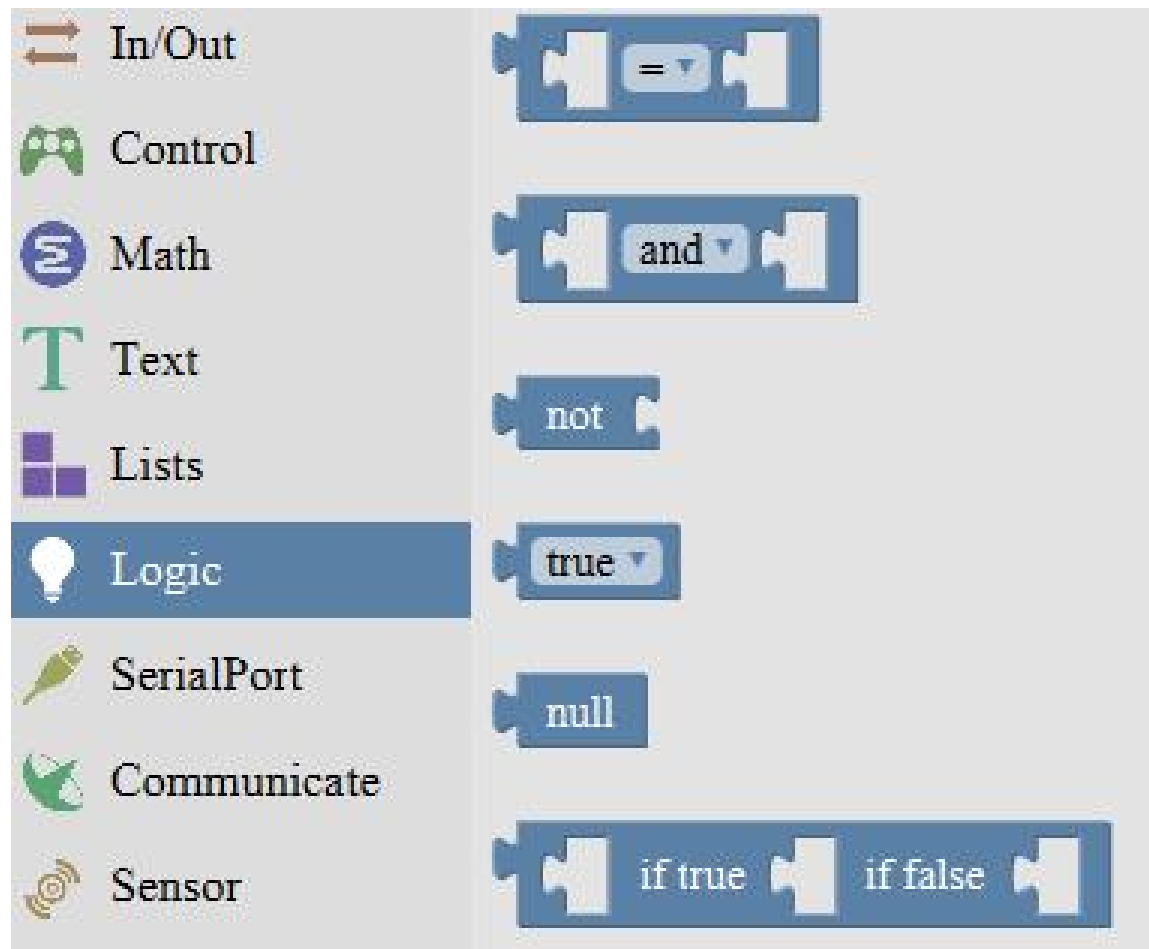
The image shows a Scratch code editor with the 'Lists' category selected in the left sidebar. The code blocks are as follows:

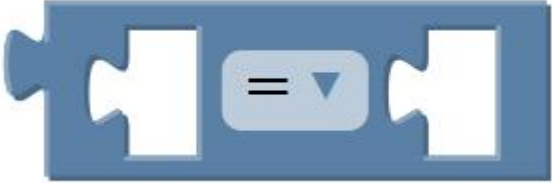
- Initialize List:** A block with a gear icon, a dropdown menu set to 'int', and a text field containing 'mylist'. Below the text field is the label 'create list with'.
- Create List from Text:** A block with a dropdown menu set to 'int', a text field containing 'mylist', and the text 'make list from text' followed by a text field containing '0,0,0' enclosed in double quotes.
- Get Length:** A block with the text 'length of' followed by a text field containing 'mylist'.
- Get Item:** A block with a text field containing 'mylist', the text 'get item at', and a numeric input field containing '1'.
- Set Item:** A block with a text field containing 'mylist', the text 'set item at', a numeric input field containing '1', the text 'to', and an empty numeric input field.

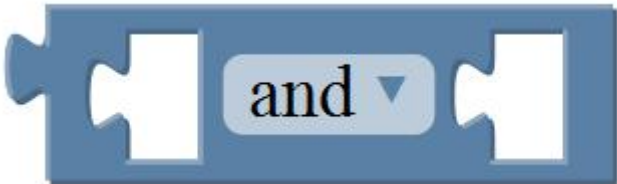
NO.	BLOCK ICON	DEFINITION
1		Create a list with any number of items
2		Creates a list from a text. (int mylist []={0,0,0};)
3		Returns the length of a list

4	 A purple Scratch block with a tab on the left and a connector on the right. It contains the text 'mylist get item at' followed by a light blue square containing the number '1'.	Returns the value of at the specified position in a list.
5	 A purple Scratch block with a tab on the left and a connector on the right. It contains the text 'mylist set item at' followed by a light blue square containing the number '1', then the text 'to' followed by a white square with a black border.	Sets the value of at the specified position in a list. Set the first item in mylist to another item.

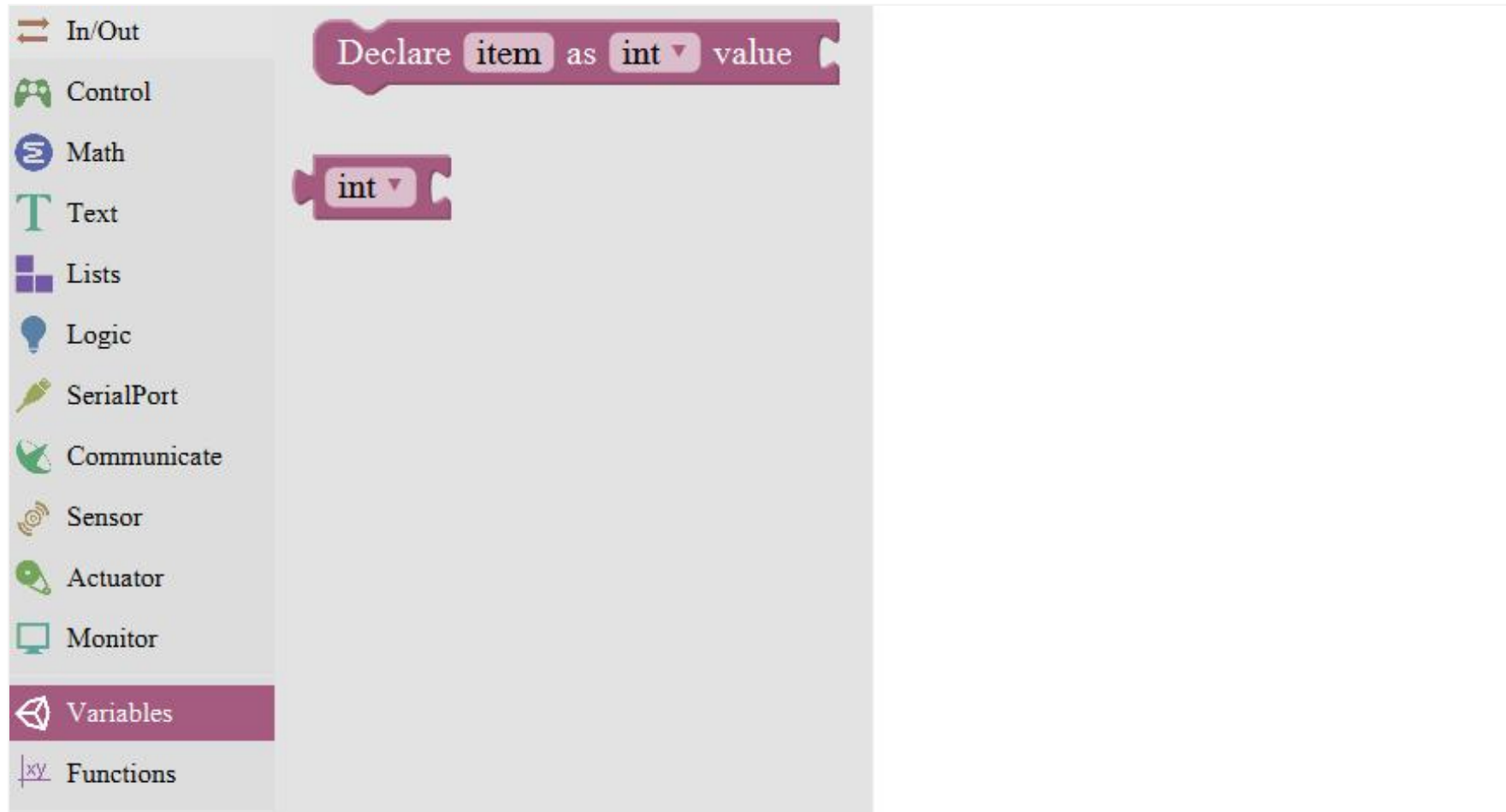
3.7 Logic Block



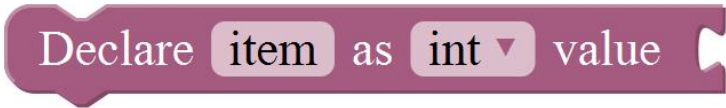
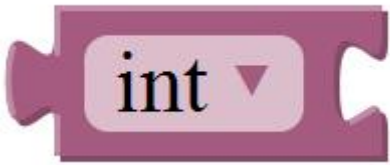
NO.	BLOCK ICON	DEFINITION
1		logic comparision =: Return true if both inputs equal each other. ≠ : Return true if both inputs are not equal to each other. <: Return true if the first input is smaller than the second input. ≤ : Return true if the first input is smaller than or equal to the second input. >: Return true if the first input is greater than the second input. ≥ : Return true if the first input is greater than or equal to the second input.

2		and: Return true if both inputs are true; or: Return true if at least one of the inputs is true
3		Returns true if the input is false. Returns false if the input is true.
4		Returns either true or false.
5		Returns null
6		If the first number is true, the second number is returned, otherwise the third number.

3.8 Variable Block

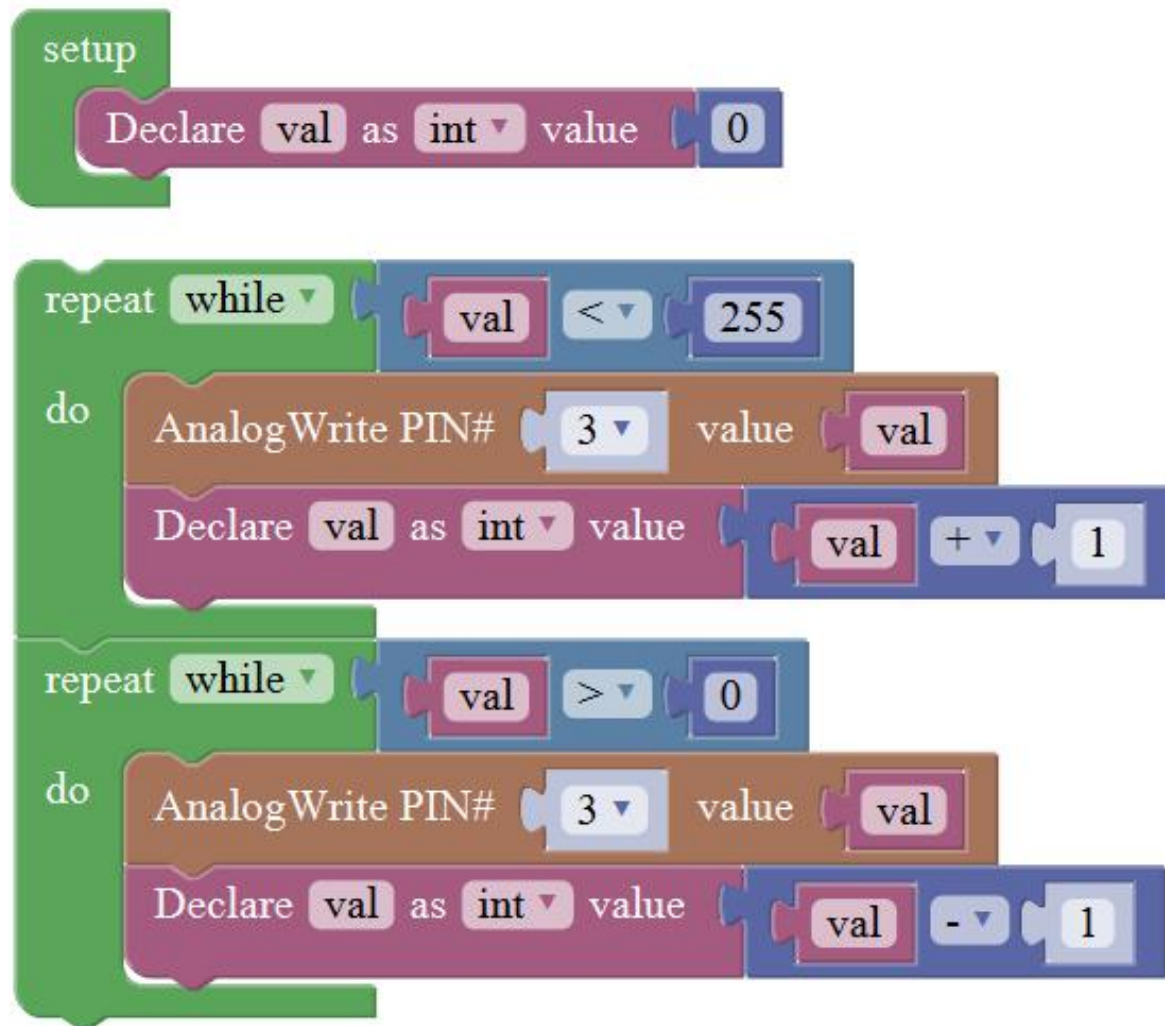


The image shows the Scratch interface. On the left is a vertical palette of category icons: In/Out, Control, Math, Text, Lists, Logic, SerialPort, Communicate, Sensor, Actuator, Monitor, Variables (highlighted in purple), and Functions. The main workspace is divided into two sections. The top section, under the 'Variables' category, contains a purple block labeled 'Declare' with a dropdown menu showing 'item', followed by 'as', another dropdown menu showing 'int', and then 'value'. Below this is a smaller purple block labeled 'int' with a dropdown arrow. The bottom section of the workspace is a large, empty white area.

NO.	BLOCK ICON	DEFINITION
1	 A Scratch 'Declare variable' block. It is a purple block with a tab on the left and a bump on the right. The text inside reads 'Declare' followed by a text input field containing 'item', then 'as', followed by a dropdown menu showing 'int' with a small downward arrow, and finally 'value'.	Declare and initialize a variable. Click to select <u>int</u> , <u>long</u> , <u>float</u> , <u>boolean</u> , <u>byte</u> , <u>char</u> , <u>string</u>
2	 A Scratch 'Data type' block. It is a purple block with a tab on the left and a bump on the right. The text inside reads 'int' followed by a small downward arrow.	Define the data types

For example: LED breath

You need an Arduino Uno and one LED module. Connect the control pin of LED module to Pin 3 of Uno board (or other pins with “~”, that is, those pins can output PWM signal). LED will gradually light then gradually dim, repeatedly.

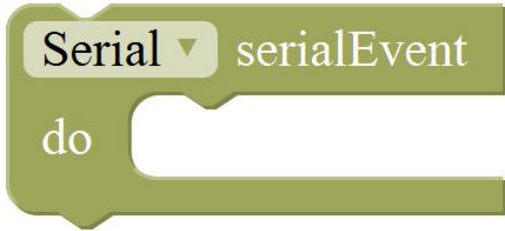


3.9 SerialPort Block



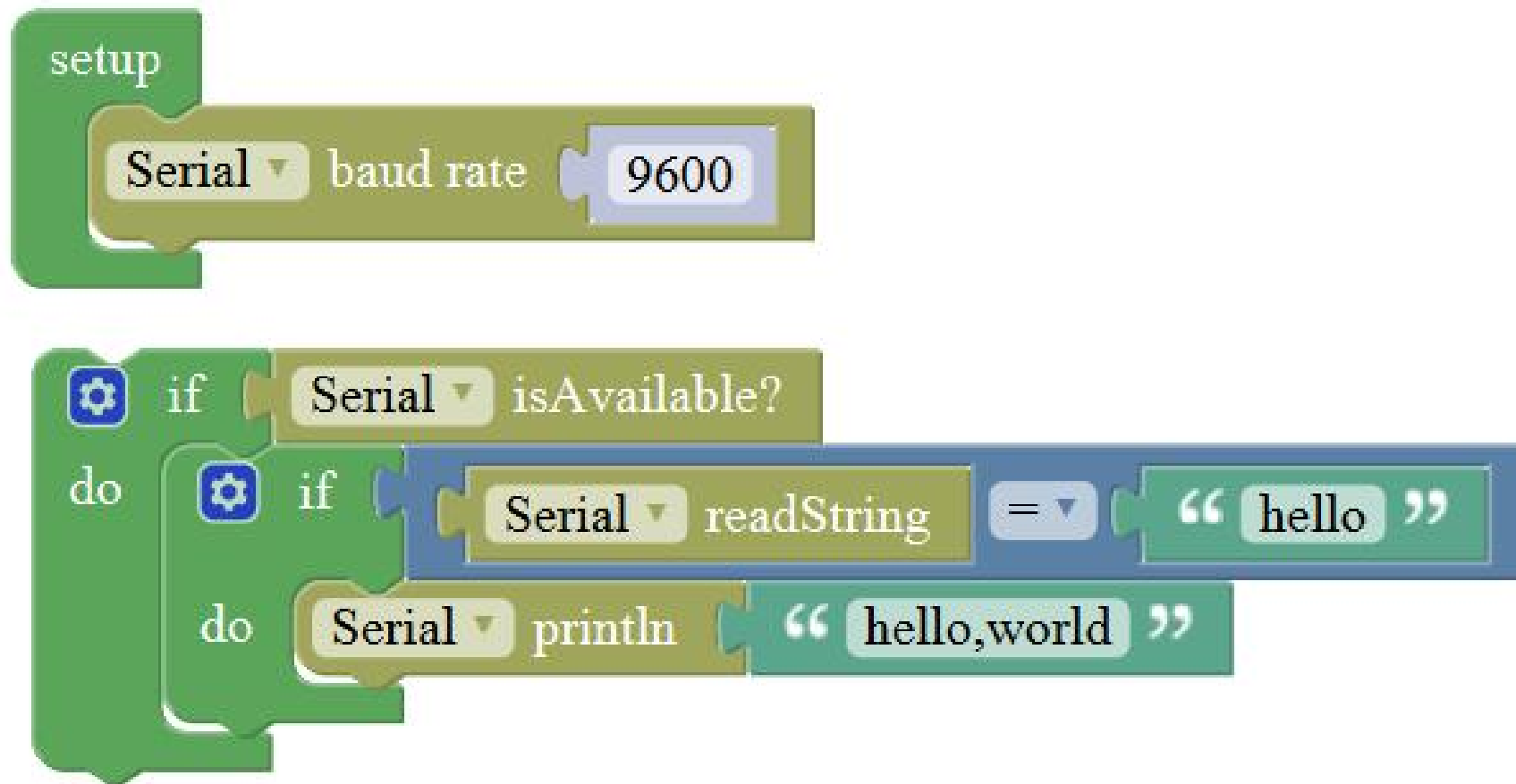
NO.	BLOCK ICON	DEFINITION
1	 The block icon for 'Serial baud rate' is a green block with a 'Serial' dropdown menu, the text 'baud rate', and a numeric input field containing '9600'.	Set the serial buad rate to 9600
2	 The block icon for 'Serial write' is a green block with a 'Serial' dropdown menu and the text 'write'.	Write the specified number, text or other value.
3	 The block icon for 'Serial print' is a green block with a 'Serial' dropdown menu and the text 'print'.	Print the specified number, text or other value on monitor.
4	 The block icon for 'Serial println' is a green block with a 'Serial' dropdown menu and the text 'println'.	Print the specified number, text or other value on newline of monitor.

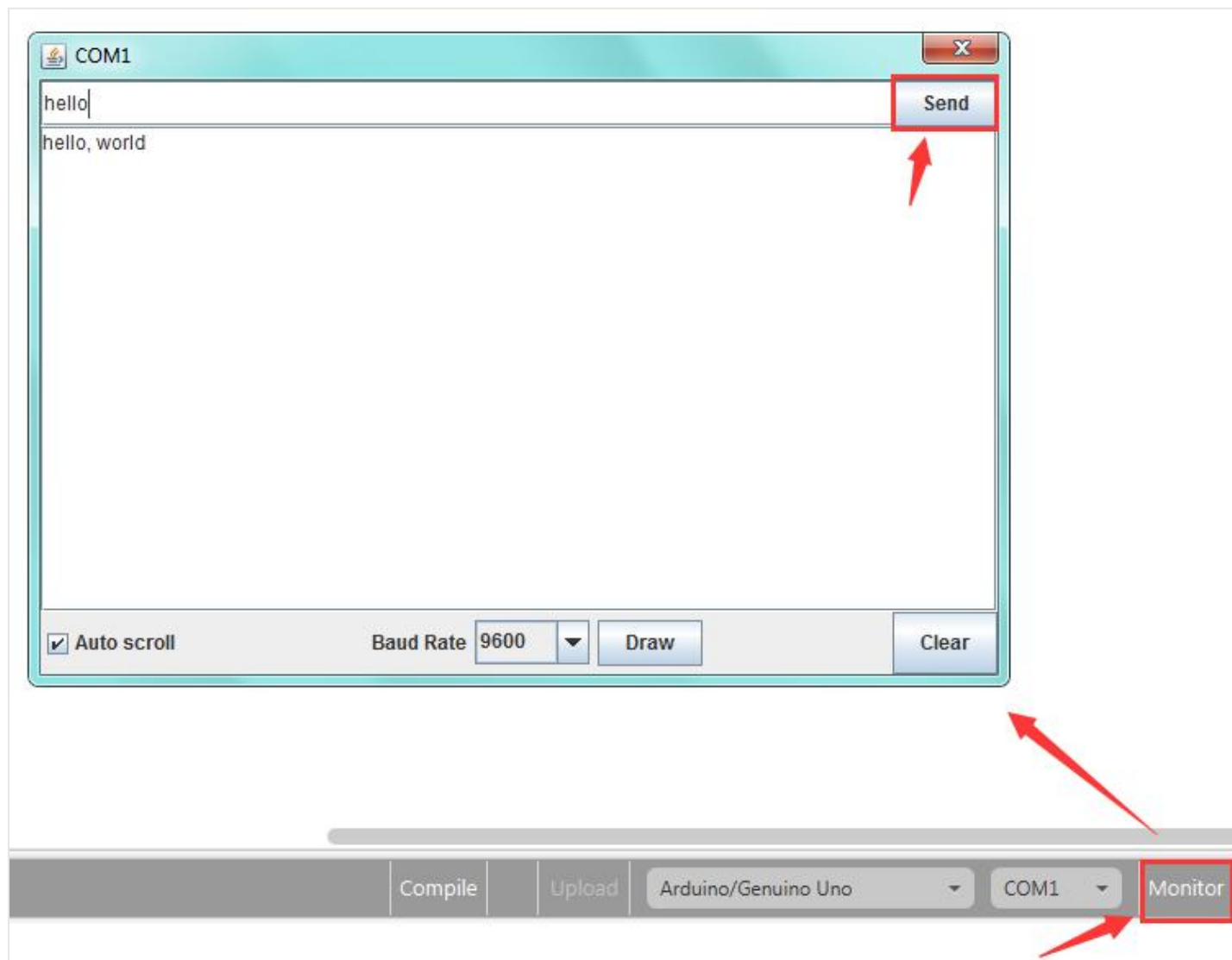
5		Print the specified number in hexademical format on newline of monitor.
6		If the serial port is available, it returns true, otherwise returns false. (generally used in Bluetooth communication)
7		Returns a string in serial port
8		A string read from serial port to a string variable, pause until read the specified character.
9		Read the serial data by byte (generally used to read the value sent from Bluetooth) (delete the data has been read)

10		Wait for the output data completed
11		Set the software serial port (call this function if need to use several serial ports)
12		Event function trigger by serial port data, that is, serial port is ready to call this function. (equal to an interrupt function)

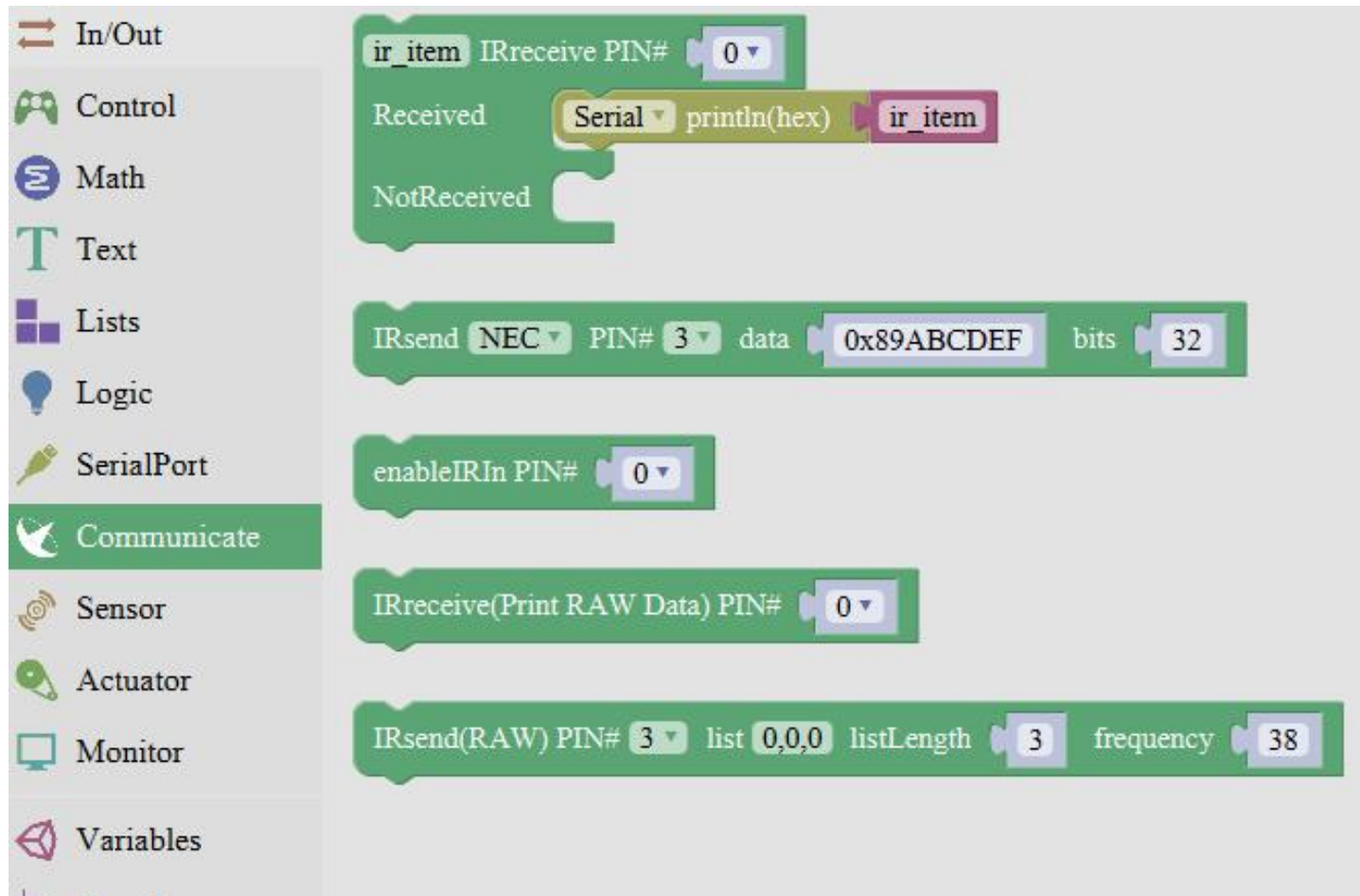
For example: serial communication

Done uploading the code, open the Arduino monitor, then enter a “hello” on the top bar, and click Send, it will print out “hello,world”.



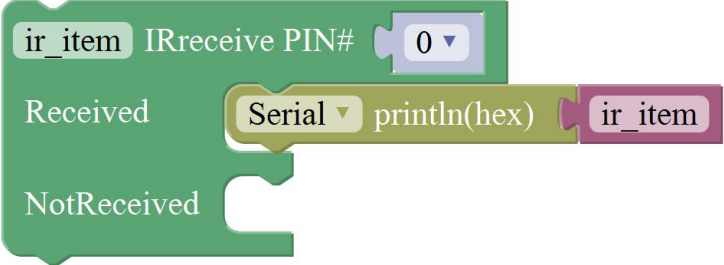
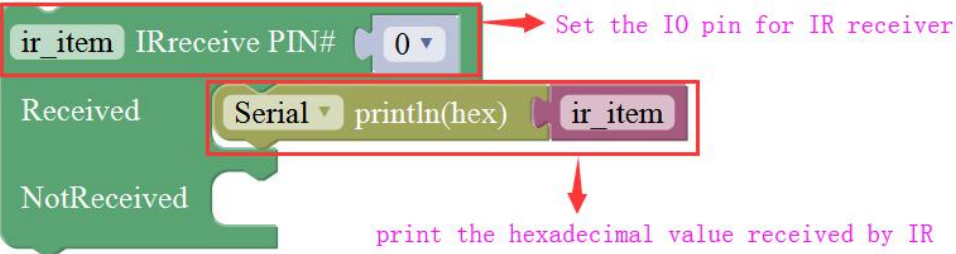


3.10 Communicate Block



The image shows a Scratch code editor with a left sidebar containing various block categories. The 'Communicate' category is highlighted in green. The main workspace contains several code blocks:

- IRreceive PIN#** block: A green block with a dropdown menu set to '0'.
- Received** block: A green block with a dropdown menu set to 'Serial' and a 'println(hex)' block.
- NotReceived** block: A green block with a dropdown menu set to '0'.
- IRsend** block: A green block with a dropdown menu set to 'NEC', a 'PIN#' dropdown set to '3', a 'data' field set to '0x89ABCDEF', and a 'bits' dropdown set to '32'.
- enableIRIn PIN#** block: A green block with a dropdown menu set to '0'.
- IRreceive(Print RAW Data) PIN#** block: A green block with a dropdown menu set to '0'.
- IRsend(RAW) PIN#** block: A green block with a 'PIN#' dropdown set to '3', a 'list' field set to '0,0,0', a 'listLength' dropdown set to '3', and a 'frequency' dropdown set to '38'.

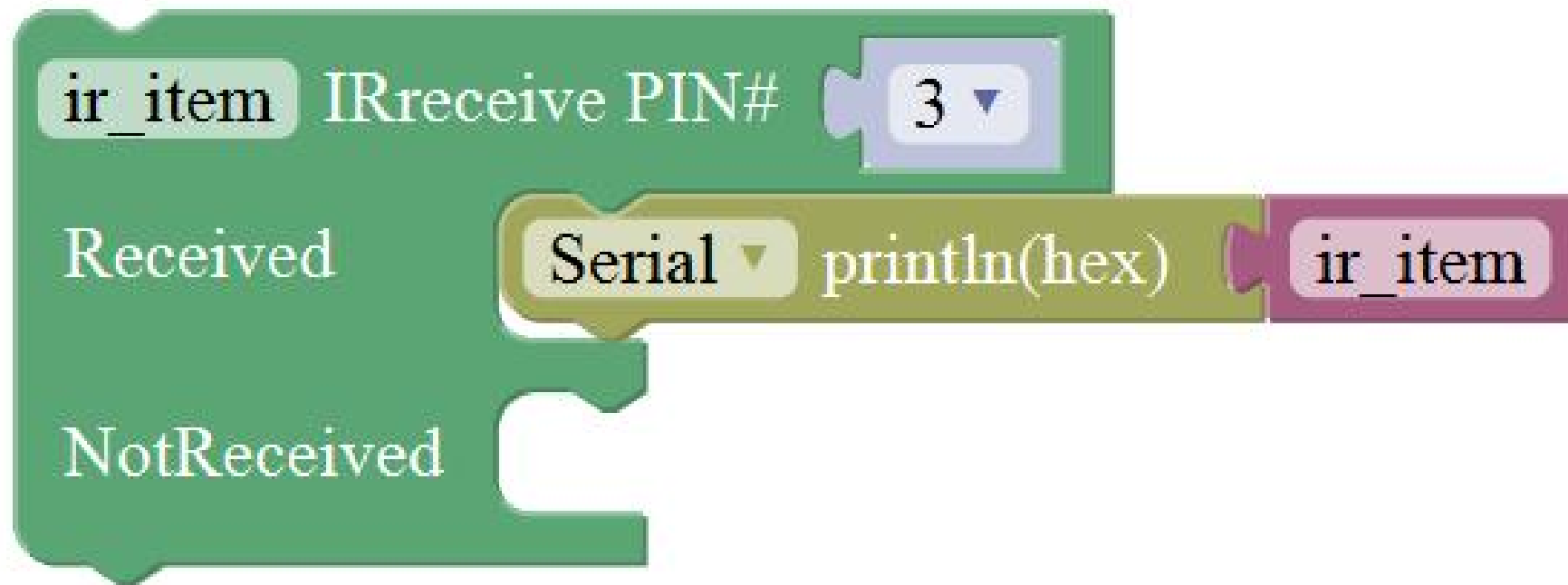
NO.	BLOCK ICON	DEFINITION
1		Do something when receiving infrared signals. 
2		Sends infrared signals of the specified types. IR transmitter sends the data, here use the libraries, only PIN3 port.

3		Enable IR decoding
4		Print the Infrared signal in RAW types when receiving it.
5		Sends RAW infrared signals (set the pin number, list, length of list and IR frequency)

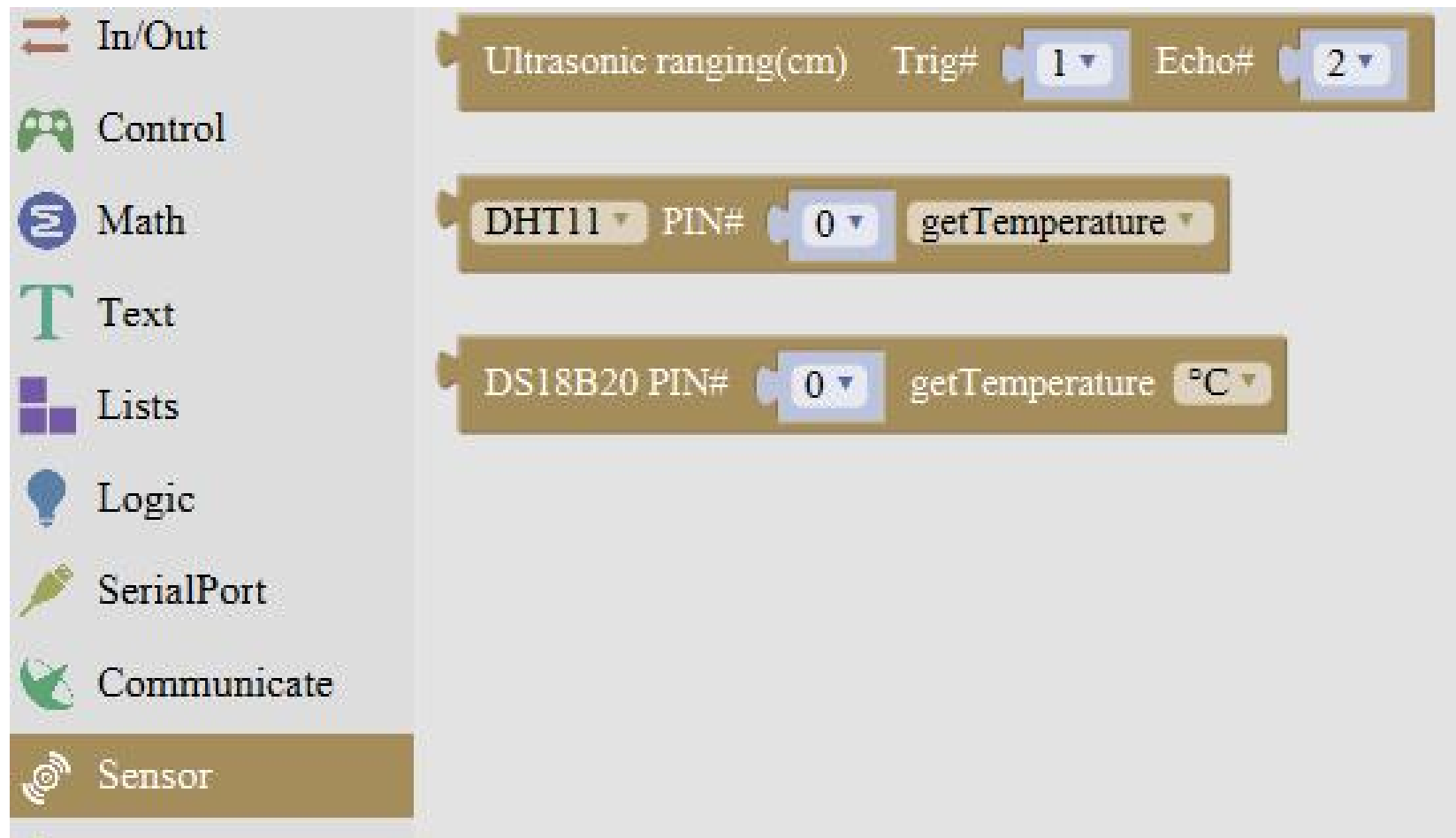
For example:

You need an Arduino Uno board, an IR receiver module and an IR remote control.

Connect the signal pin of IR receiver to Digital pin 3 of Uno board, then upload the code and open the monitor. If send a signal to an IR receiver module using an IR remote control, you should see the monitor show the corresponding signal data.

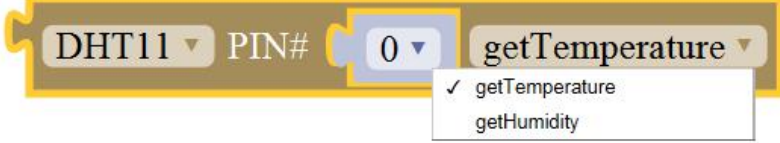


3.11 Sensor Block



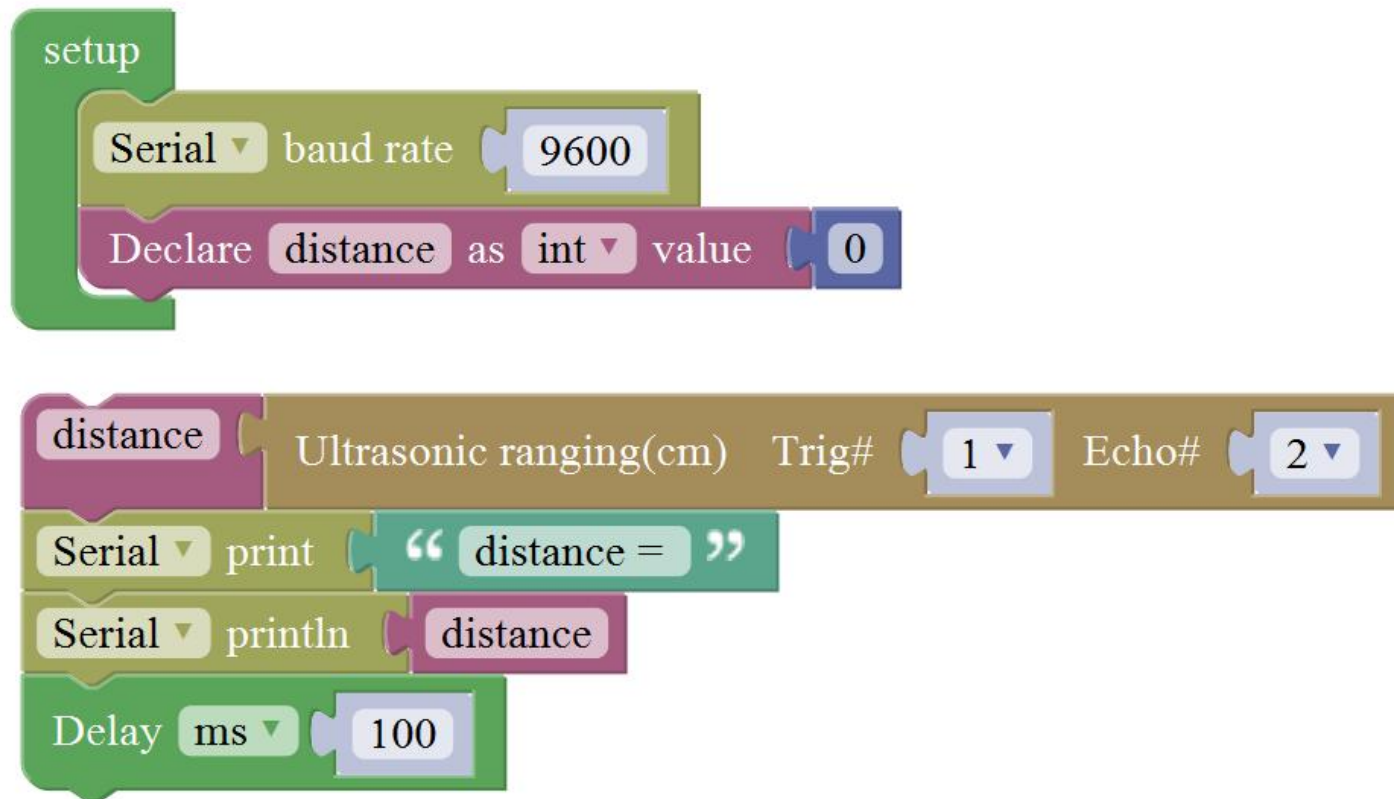
The image shows the Scratch 'Sensor' block palette on the left and three example blocks on the right. The palette includes categories: In/Out, Control, Math, Text, Lists, Logic, SerialPort, Communicate, and Sensor. The example blocks are:

- Ultrasonic ranging(cm) Trig# 1 Echo# 2
- DHT11 PIN# 0 getTemperature
- DS18B20 PIN# 0 getTemperature °C

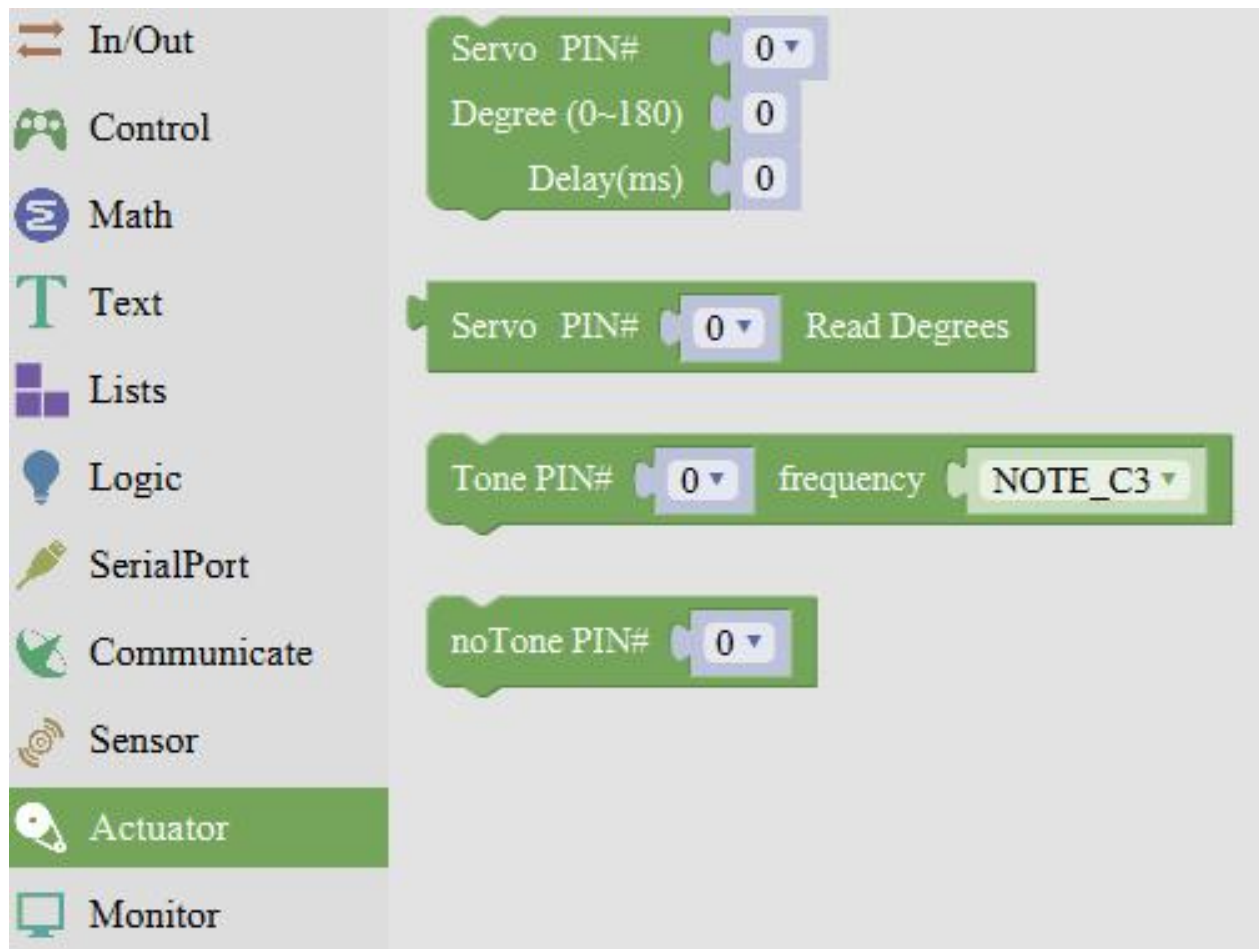
NO.	BLOCK ICON	DEFINITION
1		Set the Trig and Echo pin of ultrasonic sensor. Returns the distance of ultrasonic sensor measured. (unit: cm)
2		Set the control pin of DHT11 temperature and humidity sensor. Returns the temperature or humidity of DHT 11 sensor measured.
3		Set the pin of digital temperature sensor DS18B20. Returns the temperature value of DS18B20 sensor measured.

For example: ultrasonic ranging

Connect the Trig pin of ultrasonic sensor to Digital 1 of Uno, Echo pin to D2, then upload the code and open the monitor, you should see the distance value, updating once per 100ms.



3.12 Actuator Block



The image shows the Scratch Actuator block palette on the left and four example blocks on the right. The palette includes categories: In/Out, Control, Math, Text, Lists, Logic, SerialPort, Communicate, Sensor, Actuator (highlighted), and Monitor. The example blocks are:

- Servo Motor Control:** A block with three input fields: "Servo PIN#" (0), "Degree (0~180)" (0), and "Delay(ms)" (0).
- Servo Read:** A block with "Servo PIN#" (0) and the text "Read Degrees".
- Buzzer:** A block with "Tone PIN#" (0) and "frequency" (NOTE_C3).
- noTone:** A block with "noTone PIN#" (0).

NO.	BLOCK ICON	DEFINITION
1	 The block is a green rectangle with three input fields on the right. The first field is labeled 'Servo PIN#' and has a dropdown menu showing '0'. The second field is labeled 'Degree (0~180)' and has a dropdown menu showing '0'. The third field is labeled 'Delay(ms)' and has a dropdown menu showing '0'.	Sets the servo pin; Moves between 0-180 degree; Delay time for servo to rotate.
2	 The block is a green rectangle with an input field labeled 'Servo PIN#' and a dropdown menu showing '0', followed by the text 'Read Degrees'.	Returns that degree with the last servo move. Read the degree of servo connected to IO pin set
3	 The block is a green rectangle with two input fields. The first field is labeled 'Tone PIN#' and has a dropdown menu showing '0'. The second field is labeled 'frequency' and has a dropdown menu showing 'NOTE_C3'.	Set the pin and specified frequency for buzzer to play sound.
4	 The block is a green rectangle with an input field labeled 'noTone PIN#' and a dropdown menu showing '0'.	Stop playing sound

For example:

Connect the signal end of servo to Digital 0 of Uno, then upload the code below, servo will rotate 90 degrees. **Note:** Delay 100ms is the time required for servo to move.



3.13 Monitor Block

The image shows a block editor interface with a sidebar on the left and a workspace on the right. The sidebar contains the following categories: In/Out, Control, Math, Text, Lists, Logic, SerialPort, Communicate, Sensor, Actuator, Monitor (highlighted), Variables, Functions, CarControl, and saynum. The workspace contains the following blocks:

- setup LCD 1602 mylcd address 0x27
- LCD mylcd print line1 " " and print line2 " "
- LCD mylcd row 1 column 1 print " "
- LCD mylcd Clear
- RGB Light PIN# 0 Light Count 4
- RGB Light PIN# 0 Light number 1 R value 0 G value 0 B value 0
- RGB Light PIN# 0 Light number 1 (with a red light indicator)
- Digitdisplay_TM1650 Clear
- Digitdisplay_TM1650 displayString "abcd"
- Digitdisplay_TM1650 No. 1 Dot On

NO.	BLOCK ICON	DEFINITION
1		Set the IIC LCD1602 address
2		Input the value on LCD line 1 and line 2 from left to right.
3		Set the row and column of LCD to print the char
4		Clear the LCD screen
5		Set the control pin and the number of RGB light.

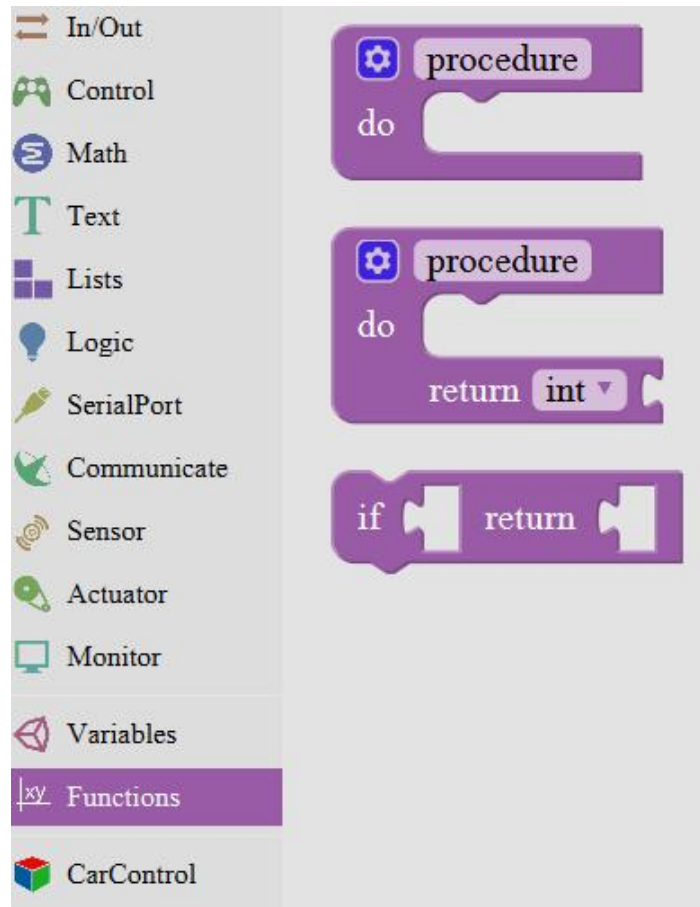
6		Set the RGB light pin, light number and brightness
7		Set the control pin, light number and color. (click to select the color)
8		Clear the data, namely turn off digital display
9		Four-digit display, displaying abcd.
10		Turn on or off the digitdisplay (here turn on the first digitdisplay)

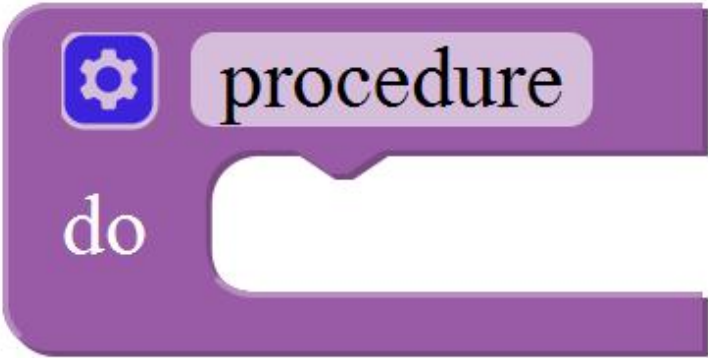
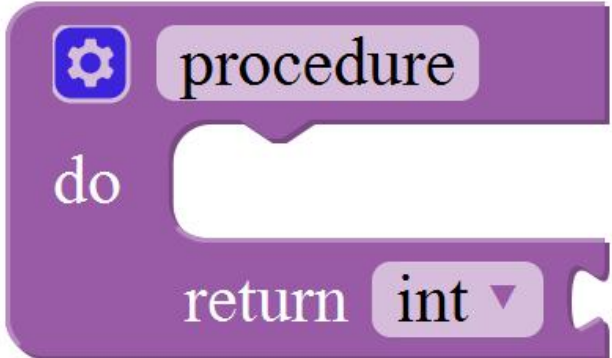
For example:

Separately connect the SDA (A4) and SCL (A5) of Arduino Uno to SDA and SCL pins of IIC LCD1602, then set the address of your LCD1602 screen, the LCD address we used here is 0x27. Then upload the code, LCD screen has two lines, you should see the line 1 print HELLO, and line 2 print 123456789.



3.14 Functions Block

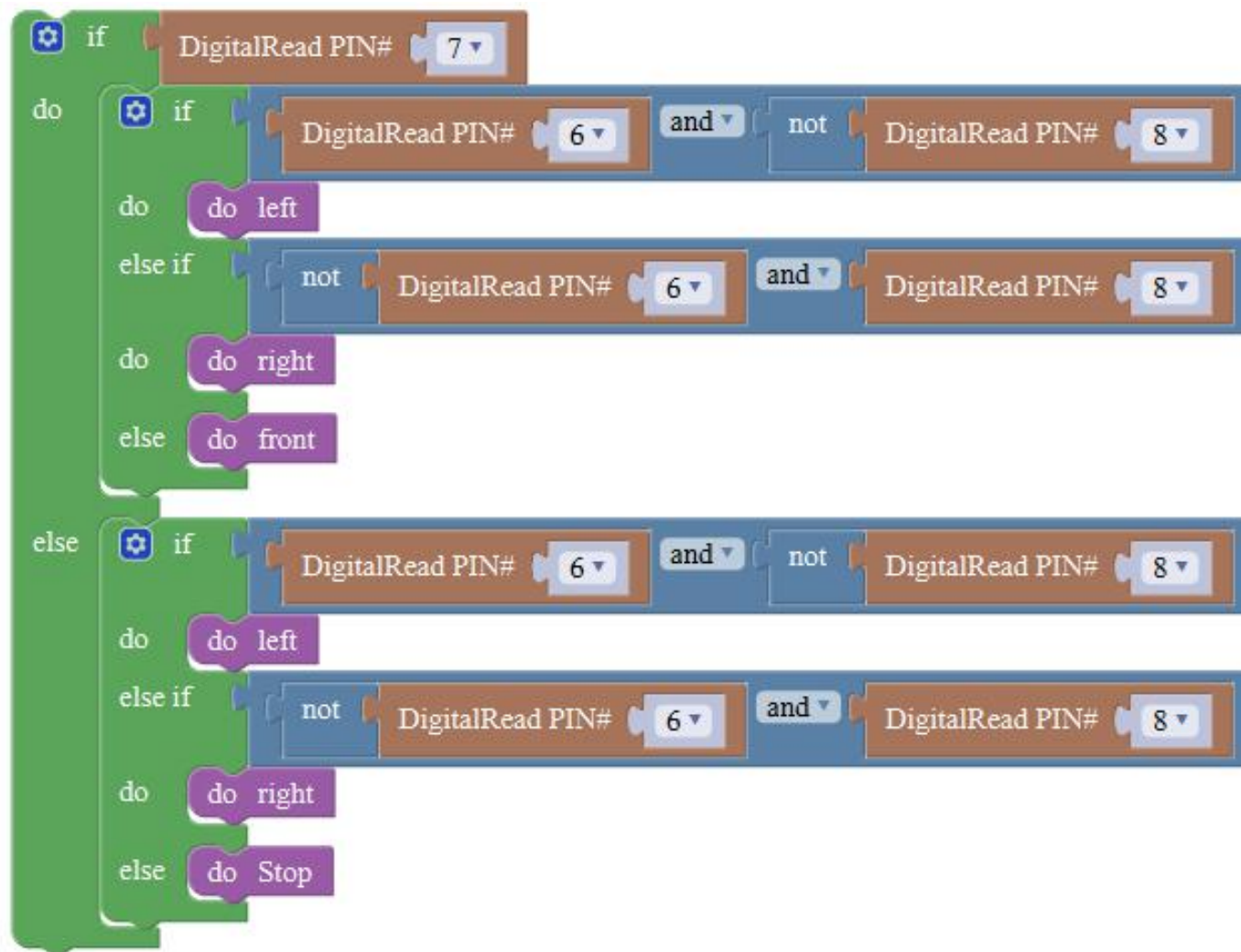


NO.	BLOCK ICON	DEFINITION
1		Creates a function with no output. Click the blue icon to set the procedure parameter. (no return value)
2		Creates a function with an output. Click the blue icon to set the procedure parameter. (with return value and can set the data types)

3		If a value is true, then return a second value.
---	--	---

For example:

Below is an example code for line tracking car. We use three tracking modules (left to D6, middle to D7, right to D8). of course you need a tracking car to test it. First edit the forward, backward, turn left, turn right and stop into functions block. Then compile and upload the code below.





4. Resources

Download the Mixly software:

https://drive.google.com/open?id=1oQxF-AZ0Aw6OQhu_8NSvwo3L2OP0Z6cU

Download the Example Code:

https://drive.google.com/open?id=1Fjd3SHHkg_-0IdB6GPX2uuTv3aRGMCSd